Report Assignment 1 ORM

Maarten van der Zwaan 2053354
Dico de Gier 2058017
project A2: lecturer R. Sotirov

24 March 2023

1 Lagrangian Relaxation

Our original problem is:

$$min \sum_{i,j} c_{ij} x_{ij}$$

$$s.t.\ x \in N,\ x \in D$$

The Lagrangian relaxation of this problem is:

$$min \sum_{i,j} c_{ij} x_{ij} + \lambda^T (b - Ax)$$

$$s.t.\ x \in N$$

where b is the right-hand-side of the constraints in D, A the constraint matrix of the constraints in D and λ the Lagrange multiplier.

In order to properly use the subgradient algorithm, we have to explain a couple of steps. Specifically we have to explain how we arrive to the equation $\lambda^{k+1} = min(\lambda^k + \mu * \gamma(x^k), 0)$. Firstly, we explain why we take the minimum. The constraints that we are going to dualize are of the form $Ax \leq b$. That means that we want to penalize our objective function when $Ax > b \Leftrightarrow 0 > b - Ax \Leftrightarrow b - Ax < 0$. The objective of the Lagrangian relaxation is to minimize, so if we want to penalize the objective function, we will have to increase the objective function. Since we already have that $b - Ax < 0$ and we want $\lambda^\intercal (b - Ax) \geq 0 \Rightarrow \lambda \leq 0$. This explains why we take the minimum of the expression and 0. Secondly, we also have to explain the '+' in the minimize expression. The aim of the Lagrangian relaxation is to minimize a certain objective function. Therefore, the aim of the Lagrangian dual will be to maximize. Since the subgradient points in the direction of steepest ascent and we want to maximize the objective function of the Lagrangian dual, we move along the subgradient. This explains the '+'.

The above explanation also explains why our initial value for lambda is the vector of all -1's, λ has to be smaller than 0. We have also observed that in the final solution most lambdas are between -1 and 0. So we believe that -1 is a good initial guess close to the real answer. Furthermore, for the stepsize μ we chose as initial estimates $\mu_0 = 1$ and $\alpha = 0.99 - p * 0.09$ where p is a penalty percentage based on the size of the problem. The reason for these choices is that both these parameters have to be 'sufficiently large', but we also want to prevent the run time from growing too large. We also use a threshold with size adjusted penalty for the objective function, so if the absolute increase of objective values is less than our threshold, the algorithm stops. Combined with a chosen ε of 0.001 plus a percentage based penalty, these parameters appear to arrive close to the correct solution. The reason for this conclusion is that we also calculate a lower and upper bound for the problem, z_{LP} and z_{IP} respectively. Since for some instances, the value of both bounds and our solution is the same, we could reasonably conclude our solution is correct.

2 Different values for k2, k3 and E

In this section, we explore the effect of different values of k2, k3 and E on the objective value of the Lagrangian dual. First of all, when our code worked for the first time, we made another observation

apart from the values of k2, k3 and E. We noticed that for constraints in D where the size of the subset is large, the corresponding lambda was nonzero i.e. the constraint was binding. By using large subsets for the E constraints, the objective value of the Lagrangian dual increased significantly. Our algorithm fills D with as many constraints with the maximal size of the subset as possible.

With the above considerations in mind, we first made a '4D' plot of TSP1 (see figure 1). We first chose TSP1 because the size of the problem ensures that we can plot in greater detail within a reasonable amount of time. In this plot we put k2 on the x-axis, k3 on the y-axis and E on the z-axis. The color of the observation then represents the value of the objective function. In figure 1, we notice the following. If we look along the k2-axis, the value of the objective function keeps increasing for arbitrary values of k3 and E. This is in contrast to the k3-axis, where the value of the objective problem stays roughly the same. This suggests that the value of k3 is less important in TSP1. If we finally look along the E-axis, we again see that the objective value increases with the value of E. We must however note that in order for the objective value to increase with E, we need a considerable increase in E, whereas the increase in k2 can be much lower. This suggest that for TSP1, if one wants to achieve the highest objective value with the lowest possible bounds, one should pay attention to k2 and E whereas k3 can almost be ignored.

Of course the above paragraph merely describes the findings for TSP1 and TSP1 is characterized by a low size (n=17). If we were to increase the size of the matrix C, one could obtain different results. However, creating such '4D' plots for larger instances like TSP6 or TSP5 is going to take forever. Therefore we decided to make another '4D' plot for TSP3 to check what happens for larger values of k2, k3 and E.

When looking at the '4D' plot generated for the larger problem TSP3 (figure 2) we see different results from TSP1. For lower values of E it pays off to both increase k2 and k3. At the highest values of E this is no longer the case as only increasing k2 yields an increase in the objective value of the Lagrangian dual. Increasing k2 leads generally for fixed values of k3 and E to higher values of the Lagrangian dual. Increasing k3 on the other hand does lead to relatively lower increase in the value of the Lagrangian dual for fixed values of k2 and E with respect to k2. Given that larger values of E will slow down the algorithm (because of the lambda that has to be found), it seems reasonable to first increase k3 to reach a higher objective value. A difference between the two instances of the TSP is that with maximum values of k2 and k3, but an E of zero, we already have reached the maximum objective value for the Lagrangian dual in TSP. When looking solely at this graph we would conclude that in order to get the best value in the shortest amount of time you would only need to increase k2 and k3, this however is not the case for all instances of the TSP as evidenced by our plot of the TSP1 test instance

We tried to generate some insightful plots for the highest sizes TSP5 and TSP6. However, we soon encountered that this would take a lot of time. In order to decrease the amount of time necessary, we tried to lower the step size on the axes, but even with very low step sizes, the amount of time necessary to make a plot was still very large. Furthermore, the resulting graph was not very insightful because of the low number of steps. We therefore decided to restrict the analysis of k2,k3 and E to TSP3. It must also be noted that for larger values of k2, k3 or E we encounter a problem where the constraint matrix being generated for the problem is larger than the memory of the device resulting in an error.

All in all, we make the following conclusions. Through our analysis of k2,k3 and E we concluded that we should take k2 as high as possible, since this gives relatively high increases within a reasonable amount of time. After k2, the difference in k3 and E is not very large. We still recommend to first increase k3, because larger values of E will cause the algorithm to run for a longer time.

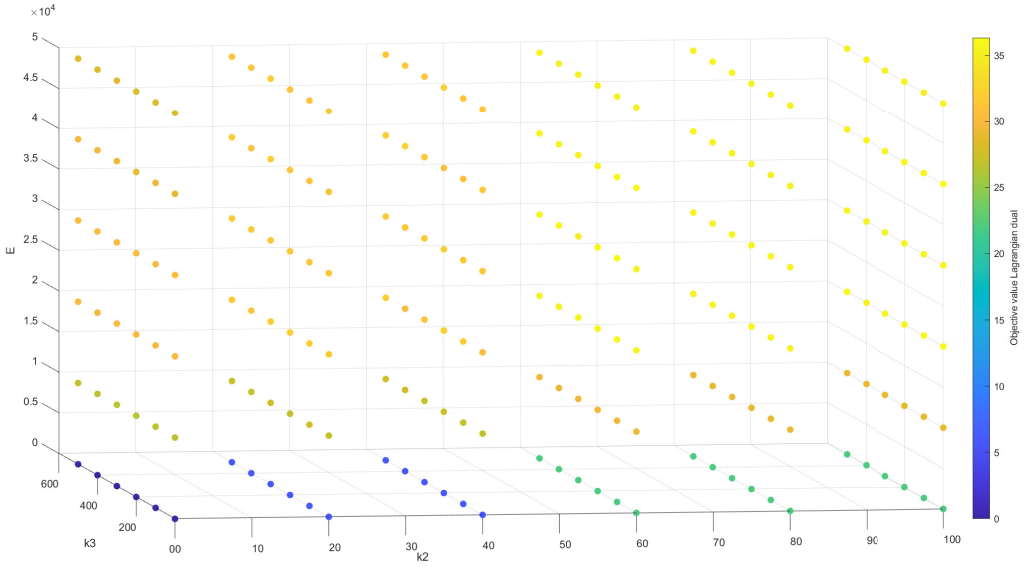

Figure 1: Objective value of TSP1 under different values of k2, k3 and E

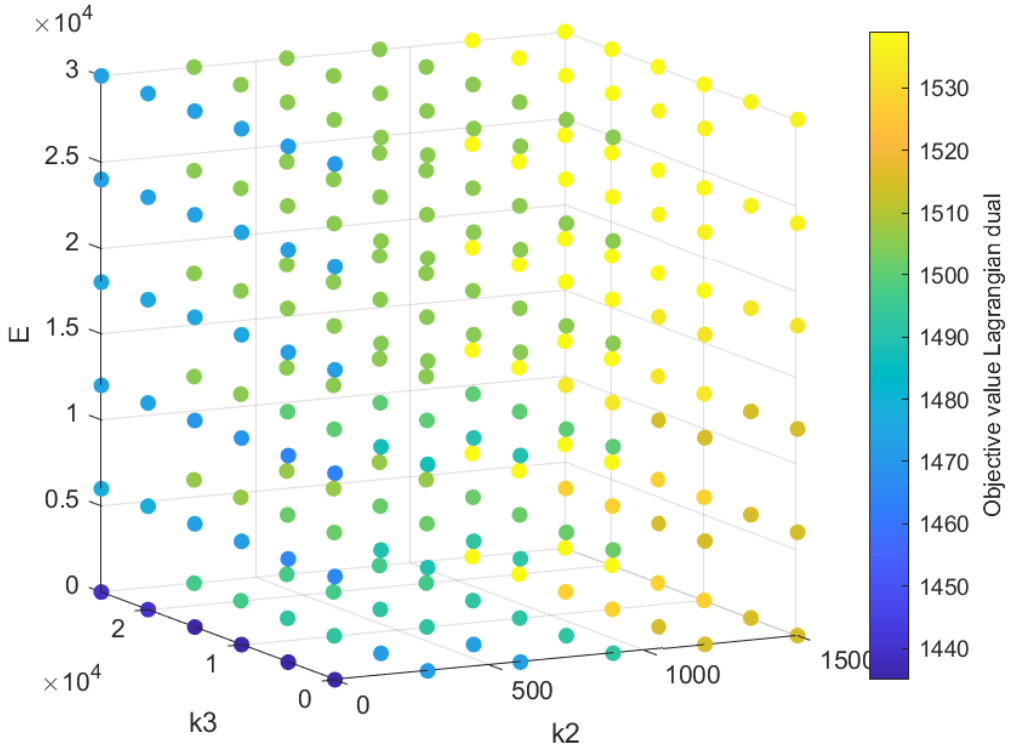

Figure 2: Objective value of TSP3 under different values of k2, k3 and E